Accelerating MPI Collectives with FPGAs in the Network and Novel Communicator Support

Qingqing Xiong*, Chen Yang*, Pouya Haghi*, Anthony Skjellum†, Martin Herbordt*
*Department of Electrical and Computer Engineering, Boston University, Boston, MA, USA
†Simcenter & Department of Computer Science and Engineering, University of Tennessee at Chattanooga
Email: *{qx, cyang90, haghi}@bu.edu, †tony-skjellum@utc.edu, *herbordt@bu.edu

***Abstract*—MPI collective operations can often be performance killers in HPC applications; we seek to solve this bottleneck by offloading them to reconfigurable hardware within the switch itself, rather than, e.g., the NIC. We have designed a hardware accelerator MPI-FPGA to implement six MPI collectives in the network. Preliminary results show that MPI-FPGA achieves 10× speedup in the most likely scenarios over conventional clusters. We introduce a novel mechanism that enables the hardware to support a large number of communicators of arbitrary shape, and that is scalable to very large systems. MPI-FPGA is fully integrated into MPICH and so transparent to MPI applications.**

High performance computing (HPC) applications often rely on collective communication for performing operations that require interaction among multiple processes; collectives comprise a large fraction of total HPC communication, and in many applications they bottleneck performance. In MPI implementations, much support has been added, which greatly complicates the software stack.

In this work [1]–[3] we offload MPI collectives into FPGA hardware (MPI-FPGA) [4]; in particular, into logic appended to the communication switches [5]–[7]. This has at least five benefits. First, it removes those extra layers of software; second, the hardware implementations are at least an order-of-magnitude faster than the software; third, it frees up the processor for other work; fourth, it distributes the execution of collective computation throughout the network, rather than forcing it into source (for broadcast) or destination (for reduction); and fifth, it reduces network load as messages generally only travel a single hop before being merged or duplicated.

Previous work in offloading collective support into hardware has been mostly limited to processing in the NIC, which has obvious limitations. General compute-in-the-network has been much studied; however, there appear to be just two recent commercial versions of in-switch computing: the IBM BlueGene family and certain switches from Mellanox. Both of these have limitations that are, in part, the result of being ASIC-based and so having strictly bounded capabilities.

MPI-FPGA has several inherent advantages: first, it is not limited to a small, fixed set of operations; second, for any application, it only needs to implement the operations that are substantially used; third, support can be extended beyond simple datatypes to higher order structures such as vectors, matrices, etc.; and fourth, compute-in-the-network can be generalized still further to support *altruistic* computing.

In this work we introduce an in-switch design capable of efficiently supporting communicators and the collectives that run on them. The first major contribution is the design, implementation, and evaluation of a set FPGA in-switch MPI collectives. We believe this to be the first FPGA version to be fully integrated into a general router. Also, MPI-FPGA is fully integrated into MPICH with publicly available code and API; MPI-FPGA is therefore currently transparently usable by any MPI application. It is also easily extended to support additional collectives or integrated into other MPI implementations. The second major contribution is the finding that all collective routing decisions–including those with arbitrarily complex communicators–only need a small amount local information.

Our experiments show that MPI-FPGA can achieve 6× - 15× speedups for MPI collectives over a CPU cluster for medium sized messages, with greater speedup for smaller messages and less for larger messages. In addition, there is little added cost over the general router itself and the enhanced router only takes a small fraction of the total device resources.

ACKNOWLEDGMENTS

This work was supported in part by the NSF through awards CCF-1562659, CCF-1562306, CCF-1618303, CCF-1617690, CCF-1822191, CCF-1821431, and CCF-1919130; by the NIH through awards 1R41GM128533 and R44GM128533; and by a grant from Red Hat.

REFERENCES

[1] J. Stern, Q. Xiong, J. Sheng, A. Skjellum, and M. Herbordt, "Accelerating MPI_Reduce with FPGAs in the Network," in *Proc Workshop on Exascale MPI*, 2017.

[2] J. Stern, Q. Xiong, A. Skjellum, and M. Herbordt, "A Novel Approach to Supporting Communicators for In-Switch Processing of MPI Collectives," in *Proc Workshop on Exascale MPI*, 2018.

[3] Q. Xiong, P. Bangalore, A. Skjellum, and M. Herbordt, "MPI Derived Datatypes: Performance and Portability Issues," in *Proceedings of the EuroMPI Conference*, 2018.

[4] A. George, M. Herbordt, H. Lam, A. Lawande, J. Sheng, and C. Yang, "Novo-G#: A Community Resource for Exploring Large-Scale Reconfigurable Computing Through Direct and Programmable Interconnects," in *IEEE High Perf. Extreme Computing Conf.*, 2016.

[5] J. Sheng, C. Yang, and M. Herbordt, "Application-Aware Collective Communication on FPGA Clusters," in *Proc. IEEE Symposium on Field Programmable Custom Computing Machines*, 2016.

[6] J. Sheng, Q. Xiong, C. Yang, and M. Herbordt, "Collective Communication on FPGA Clusters with Static Scheduling," *Computer Architecture News*, vol. 44, no. 4, 2016.

[7] J. Sheng, C. Yang, and M. Herbordt, "High Performance Dynamic Communication on Reconfigurable Clusters," in *Proc. IEEE Conf. on Field Programmable Logic and Applications*, 2018.

2576-2621/20/$31.00 ©2020 IEEE
DOI 10.1109/FCCM48280.2020.00046